FPEL05150060

10/577579

Express Mail No.: EV 839874085 US

IAP12 Rec'd PCT/PTO 27 APR 2006

OPERATING MEDIA DEVICES IN PRE-OS ENVIRONMENT

FIELD

[0001] Embodiments of the invention relate to operating media devices by a computing device, and more particularly to operating media devices by a computing device in a pre-OS environment.

BACKGROUND

[0002] The essential architecture of computer systems, such as personal computers, includes a central processing unit (CPU) in communication with a system memory that includes a memory medium and a memory controller interface. In addition, a computer system typically includes display interfaces, such as graphical interfaces, for operating video displays and input/output (I/O) control logic for various I/O devices, including a keyboard, mouse, floppy drive, hard drive, etc. An operating system (OS), such as Windows®, typically stored in the memory medium, monitors and conducts operations of the computer system, such as reading instructions and data from I/O devices and system memory. The operating system is typically loaded and executed from the system memory immediately following power-up (i.e. booting).

[0003] While the operating system provides the computer system with the ability to function, it is typically not the only means of booting a computer system. An alternative method commonly known as pre-OS booting, may also be employed to boot up a computer system. As the name suggests, pre-OS booting occurs prior to the loading and execution of the main operating system. In a typical pre-OS booting, a boot image file is accessed, such as from firmware, and executed, which results in the computer system to operate in a pre-OS environment. Due to the limited instructions in the boot image file, however, the computer system's functionality during the pre-OS environment is also typically limited as compared to the functionality of the computer system during executions of a main operating system. One limitation is in the simultaneous operating of two or more media devices, such as video devices.

[0004] Currently, pre-OS firmware does not support the capability for allowing the computer system to interact with multiple devices at the same time. When multiple video devices are detected, the computer system typically selects one of the devices as the primary device and ignores all others. This is in part due to the limited decode range of the widely used video graphics array (VGA), which restricts the computer system to single VGA device usage, and in part due to computer system's inability to install and dispatch the multiple video option ROMs needed for interacting with multiple video devices at the same time.

DETAILED DESCRIPTION OF THE INVENTION

[0005] The invention may best be understood by referring to the following description and accompanying drawings that are used to illustrate embodiments of the invention.

[0006] FIG. 1 is a block diagram of a computer system in which embodiments of the invention can be practiced.

[0007] FIG. 2 is a flow chart illustrating a process according to an exemplary embodiment of the invention.

[0008] FIGs. 3A-B illustrate block diagrams of a system memory in which embodiments of the invention can be practiced.

[0009] FIGs. 4-6 are flow charts further illustrating the processes according to exemplary embodiment of the invention shown in FIG. 2.

DETAILED DESCRIPTION

[00010] Embodiments of the invention generally relate to a system and method for operating media devices by a computing device in a pre-OS environment. Herein, on embodiment of the invention may be applicable to media devices used in a variety of computing devices, which are generally considered stationary or portable electronic devices. Examples of a computing device may include, but are not limited or restricted to a computer, a set-top box, video game systems, music playback systems, and the like.

[00011] Reference in the specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the invention. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment. Some embodiments of the invention are implemented in a machine-accessible medium. A machine- accessible medium includes any mechanism that provides (i.e., stores and/or transmits) information in a form accessible by a machine (e.g., a computer, network device, personal digital assistant, manufacturing tool, any device with a set of one or more processors, etc.). For example, a machine-accessible medium includes recordable/non-recordable media (e.g., read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; etc.), as well as electrical, optical, acoustical or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.), etc.

[00012] In the following description, numerous details are set forth. It will be apparent, however, to one skilled in the art, that the embodiments of the invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form, rather than in detail, in order to avoid obscuring the embodiments of the invention.

[00013] Also in the following description are certain terminologies used to describe features of the various embodiments of the invention. For example, the term "media device" refers any on-board or plug-in device, such as video cards, music players, or DVD

players, for example, that is capable of storing video or audio data, such as movies, songs, etc. The term "linear buffer" refers to one or more buffers of a memory system in which obtained data and instructions can be stored. The term "software" generally denotes executable code such as an operating system, an application, an applet, a routine or even one or more instructions. The software may be stored in any type of memory, namely suitable storage medium such as a programmable electronic circuit, a semiconductor memory device, a volatile memory (e.g., random access memory, etc.), a non-volatile memory (e.g., read-only memory, flash memory, etc.), a floppy diskette, an optical disk (e.g., compact disk or digital versatile disc "DVD"), a hard drive disk, or tape. The term "pre-OS" (also known as pre-boot) environment refers to a setting in which tasks are performed before a main operating system (OS) is loaded, and may include limited use of a disk operating system (DOS).

[00014] With reference to FIG. 1, an embodiment of an exemplary computer environment is illustrated. In an exemplary embodiment of the invention, a computing device 100, such as a personal computer, comprises a bus 105, such as a Peripheral Component Interconnect (PCI) bus for example, or other communication medium for communicating information. A processor 111 is coupled to the bus 105 for processing information.

[00015] The computing device 100 further comprises a system memory 140 which comprises a main memory 143, such as random access memory (RAM) or other dynamic storage device as for storing information and instructions to be executed by the processor 111. Main memory 143 also may be used for storing temporary variables or other intermediate information during execution of instructions by the processor 111. The system memory 140 also may comprise a read only memory (ROM) 144 and/or other static storage devices 145 for storing static information and instructions for the processor 111, such as magnetic disk or optical disc and its corresponding drive, flash memory or other nonvolatile memory, or other memory device. Such elements may be combined together or may be separate components, and utilize parts of other elements of the computing device 100.

[00016] The system memory 140 also comprises a memory decoder 142 for translating between the addresses and storage format used by the processor 111 and the format used by memory chips and modules of memory system 140, such as main memory 115, ROM 144 and storage devices 145. The system memory 140 also comprises a memory controller 141 which controls the memory system 140 and a memory mapping logic 146 which may also be implemented in hardware or software stored in system memory 140. System memory 140 may also include other forms of storage (not shown) such as registers, caches etc.

[00017] The computing device 100 may also be implemented with a display device 130 coupled to a bus 105, such as a liquid crystal display (LCD) or other display technology, for displaying information to an end user. In some environments, the display device 130 may be a touch-screen that is also utilized as at least a part of an input device. In other environments, display device 130 may be or may include an auditory device, such as a speaker for providing auditory information.

[00018] Other devices included are media devices 160, shown as media device _1 through media device _N (N>1), which are devices capable of storing video or audio data, such as movies, songs, etc. The media device 160 communicates with the processor 111, and may further generate its results on the display device 130. A communication device 150 may also be coupled to the bus 105. Depending upon the particular implementation, the communication device 150 may include a transceiver, a wireless modem, a network interface card, or other interface device. The computing device 100 may be linked to a network or to other devices using the communication device 150, which may include links to the Internet, a local area network, or another environment. In an embodiment of the invention, the communication device 150 may provide a link to a service provider over a network. The computing device 100 also includes input/output (I/O) decoder 120, display interface decoder 135 and control logic 170 (stored in hardware or software) whose functions are described in greater detail in conjunction with FIGs. 3-6 below.

[00019] FIG. 2 is a flow chart illustrating a process according to an exemplary embodiment of the invention. The elements of the computing system 100 shown in FIG. 1 are referenced in conjunction with FIG. 2 for illustrative purposes. As shown in FIG. 2, following an action to start the process (block 200), a plurality of media

devices 160 in communication with the computing device 100 are selected (block 210) following detection. Suitably at least one of the media devices 160 comprises an on-board device and a plug-in device, such as video devices (e.g., a video cards), audio devices (e,g, digital music players), and audio/video devices (e.g., DVD players). Next, communication resources, such as PCI bus resources, for the media devices 160 selected by the computing device are allocated and programmed (block 220).

[00020] The selected media devices 160 are then initialized (block 230), each at a different time period, as described in greater detail in conjunction with FIGs. 3-6 below. The information corresponding to each initialized media device 160 is then mapped (block 240) to different memory locations of the computing device 100 by the memory mapping logic 146, as described in greater detail in conjunction with FIGs. 3-6 below. Next, the computer system 100 operates, such as interacts with, the initialized media devices 160 based on the mapped information corresponding to each operated media device 160 (block 250). The operating of the initialized media devices 160 is performed while the computing device is in a pre-OS environment. The process then ends (block 260).

[00021] The overall operations of the FIG. 2 will now be illustrated in further detail in conjunction with exemplary circuit block diagrams of FIGs. 3A-B and flow charts of FIG. 4-6. Referring to FIGs. 3A-B, for simplicity only two media devices 160, such as media device_1 and media device_2 are shown although embodiments of the invention are not limited to only two media devices 160. As shown in FIGs. 3A-B, a memory region 147 is selected by the system memory 140, such as in the main memory 143 or the storage device 145. The memory region 147 comprises addresses, such as from 0 to 5 GB. At the lower end of the addresses, such as 0 to C0000, are the I/O region 147c and memory region 147d in which is stored I/O and general information of a video graphics array (VGA) interface, such as a Legacy VGA interface. At the upper end of the addresses, such as 3GB and higher, are a set of memory locations 147a, such as line fill buffers (LFB), such as LFB_1 and LFB_2. In an exemplary embodiment of the invention, the number of memory locations 147a

used corresponds to number of the media devices 160 selected. In the example of FIGs. 3A-B, only two line fill buffers, LFB_1 and LFB_2 are shown since only two media devices 160, media device_1 and media device_2, are selected, for simplicity.

[00022] FIG. 4 is an exemplary flow chart further illustrating the initializing and mapping processes used in FIG. 2 (blocks 230, 240) in the context of the simplified example of FIGs. 3A-B when only two media device 160, media device_1 and media device_2 are selected. As shown in FIG. 4, the process starts (block 400) and proceeds to initialize a first media device 160, such as media device_1, during a first time period (block 410) as described in greater detail in conjunction with FIG. 5 below. Information corresponding to the initialized media device_1 is then mapped (shown symbolically by line 300 in FIG. 3A) by the memory mapping logic 146 to a memory location 147a corresponding to the media device_1, such as to LFB_1 (block 420). Next, a second media device 160, such as media device_2, is initialized (block 430) during a second time period that is subsequent to the first time period corresponding to the initialization of media device_1, as described in greater detail in conjunction with FIG. 6 below. Information corresponding to the initialized media device_2 is then mapped (shown symbolically by line 305 in FIG. 3B) by the memory mapping logic 146 to a memory location 147a corresponding to the media device_2, such as to LFB_2 (block 440). The flow is then returned (block 450) to FIG. 2 (block 240).

[00023] FIG. 5 is an exemplary flow chart further illustrating the initializing process for a first media device 160 illustrated in block 410 of FIG. 4. As shown in FIG. 5, the process starts (block 500) and proceeds to enabling a decoding of a display interface on a path of the media device_1, including all upstream buses 105, such as PCI buses (block 510). In an exemplary embodiment of the invention, the display interface comprises a video graphics array (VGA) interface, and the VGA decoding on the path of the media device_1 is performed by the display interface decoder 135 (shown in FIG. 1). Next, input/output decoding is enabled for the media device_1 (block 520), such as by media device_1, such as by using the Input/Output decoder 120 shown in FIG. 1. The input/output decoding is performed on the information stored in the I/O region 147c of

FIG. 3A as described above.

[00024] A memory decoding is then enabled for the media device_1 (block 530), such as by using the memory decoder 142 shown in FIG. 1. The memory decoding is performed on the information stored in the memory region 147d of FIG. 3A as described above. Service instructions corresponding to the media device_1 are thereafter loaded and dispatched, such as by the memory controller 141 from the ROM 144 (block 540). The service instructions corresponding to the media device_1 may for instance include video service instructions, audio service instructions or both. For instance, the video service instructions may comprise option ROM instructions, such as option ROM_1 shown in FIG. 3A, stored in the ROM 144 and loaded into memory region 147b. A memory information and a mode corresponding to a memory location 147a, such as memory location LFB_1, is then obtained by the memory controller 141 (block 550). The media device_1 is thereafter switched by the control logic 170 to the mode obtained for the media device_1 (block 560). The flow is then returned (block 570).

[00025] As described above in conjunction with FIG. 4 (block 420), the information corresponding to the initialized media device_1 is mapped (shown symbolically by line 300 in FIG. 3A) to the memory location LFB_1 (obtained in block 550 of FIG. 5). The information corresponding to the initialized media device_1 may for instance include instructions and/or addresses corresponding to the media device_1 and those stored in regions 147b (such as ROM_1), 147c and 147d. Since the media device_1 was previously switched to an operation mode corresponding to the memory location LFB_1 (FIG. 5, block 560), the media device_1 will from thereon use the mapped information residing in memory location LFB_1 to interact with the computing device 100 (shown symbolically by line 301 in FIG. 3A), instead of interacting using the information stored in regions 147b, 147c and 147d (shown symbolically by line 302 in FIG. 3A).

[00026] FIG. 6 is an exemplary flow chart further illustrating the initializing process for a second media device 160 illustrated in blocks 430 of FIG. 4. As shown in FIG. 6, the process starts (block 600) and proceeds to disabling the enabled

decoding of the display interface on the path of the media device_1, as previously illustrated in block 510 of FIG. 5 (block 610). The enabled input/output decoding for the media device_1 and the enabled memory decoding for the media device_1 are then disabled (blocks 620, 630). The input/output decoding and memory decoding for the media device_2 is enabled for the media device_2 (blocks 640, 650).

[00027] Following the foregoing enablement and disablement operation of block 610-650, the service instructions corresponding to the media device_2 are loaded and dispatched by memory controller 141 (block 660). According to an exemplary embodiment, the service instructions corresponding to the media device_2 comprises video service instructions, audio service instructions or both. The video service instructions may for instance comprise an option ROM instructions, such as option ROM_2 shown in FIG. 3B, stored in ROM 144 (FIG. 1) and loaded into memory region 147b. The storage of option ROM_2 instructions in the region 147b will partially or fully overwrite the previously stored option ROM_1 instructions in the region 147b. This however, will not affect the operation of the media device_1, since as described above the information needed for the operating of media device_1 by the computing device 100 is now residing in LFB_1. Next, a memory information and a mode corresponding to another memory location 147a, such a memory location LFB_2, is obtained by the memory controller 141 (block 670) and the media device_2 is switched by the control logic 170 to the mode obtained for the media device_2 (block 680). In an exemplary embodiment of the invention, the memory decoding for the media device_1 is re-enabled by the memory decoder 142 following the initialization of media device_2. The flow is then returned (block 690) to FIG. 4 (block 430).

[00028] As described above in conjunction with FIG. 4 (block 440), the information corresponding to the initialized media device_2 is mapped (shown symbolically by line 305 in FIG. 3B) to the memory location LFB_2 (obtained in block 670 of FIG. 6). The information corresponding to the initialized media device_2 may for instance include instructions and/or addresses corresponding to media device and those stored in regions 147b (such as ROM_2), 147c and 147d. The flow is then returned (block 450) to FIG. 2

(block 240). Since the media device_2 was previously switched to an operation mode corresponding to the memory location LFB_2 (FIG. 6, block 680), the media device_2 will from thereon use the mapped information residing in memory location LFB_2 to interact with the computing device 100 (shown symbolically by line 304 in FIG. 3B), instead of interacting using the information stored in regions 147b, 147c and 147d (shown symbolically by line 303 in FIG. 3A). The regions 147b, 147c and 147d can then be freed for initialization of additional selected media devices 160 in the manner described above.

[00029] In this way, as previously described in conjunction with FIG. 2 (block 250) the computing device 100 can separately operate and interact in a pre-OS environment with each of the media device_1 and media device_2. Thus, so long as each media device 160 is initialized at a different time period and set to a mode corresponding to the initialized media device 160, then line frame buffers 147a corresponding to each initialized media device 160 can be used to interact with each initialized media device 160 at the same time in a pre-OS environment, regardless of the information stored in regions 147b, 147c and 147d.

[00030] In an exemplary embodiment of the invention, the software that, if executed by a computing device 100, will cause the computing device 100 to perform the above operations described in conjunction with FIGs. 2-6 is stored in a storage medium, such as main memory 143, and storage devices 145. Suitably, the storage medium is implemented within the processor 111 of the computing device 100.

[00031] It should be noted that the various features of the foregoing embodiments were discussed separately for clarity of description only and they can be incorporated in whole or in part into a single embodiment of the invention having all or some of these features.